Adaptive Critic Design for Aircraft Control

Silvia Ferrari

Advisor: Prof. Robert F. Stengel

Princeton University

FAA/NASA Joint University Program on Air Transportation,

MIT, Cambridge, MA

January 18-19, 2001

Table of Contents

Introduction

- Gain scheduled linear controllers for nonlinear systems

- Classical/neural synthesis of control systems
  Prior knowledge
  Adaptive control and artificial neural networks

- Adaptive critics
  Learn in real time
  Cope with noise
  Cope with many variables
  Plan over time in a complex way
  ...

Action network takes immediate control action

Critic network estimates projected cost

Motivation for Neural Network-Based Controller

- Network of networks motivated by linear control structure

- Multiphase learning:

  Pre-training

  On-line training during piloted simulations or testing

- Improved global control

- Pre-training phase provides:

  A global neural network controller

  An excellent initialization point for on-line learning

- On-line training accounts for:

  Differences between actual and assumed dynamic models

  Nonlinear effects not captured in linearizations

Nonlinear Business Jet Aircraft Model

Aircraft Equations of Motion:

$$\frac{d\mathbf{x}(t)}{dt} \equiv \dot{\mathbf{x}}(t) = \mathbf{f}\left[\mathbf{x}(t), \mathbf{p}(t), \mathbf{u}(t)\right]$$

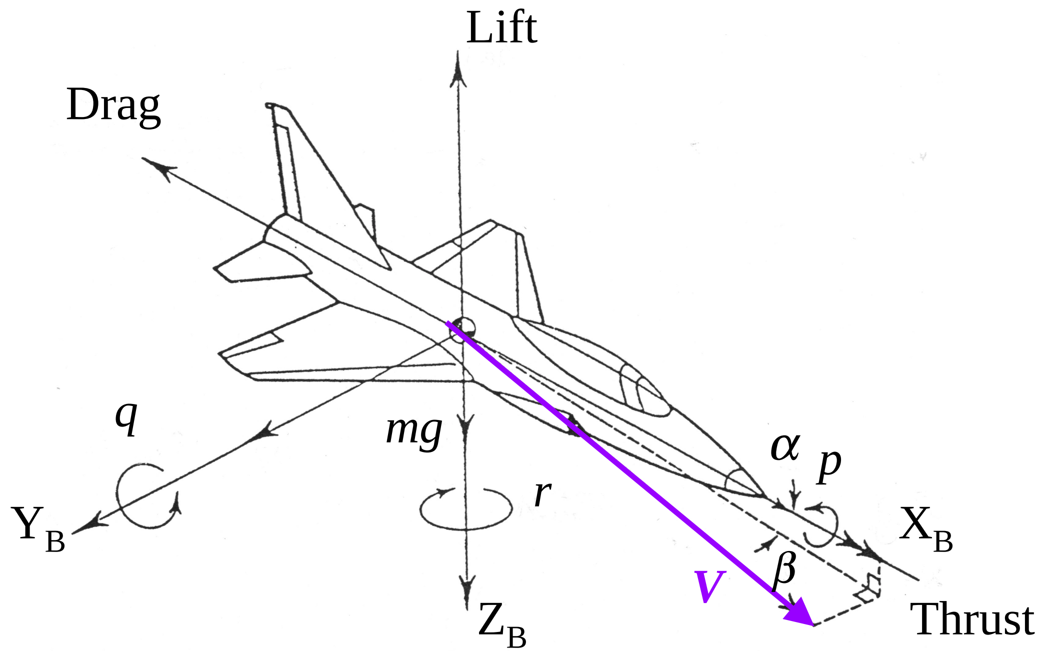
$$\mathbf{y}(t) = \mathbf{h}\left[\mathbf{x}(t), \mathbf{p}(t), \mathbf{u}(t)\right]$$

State vector: $\mathbf{x}(t) \in \Re^n$

Control vector: $\mathbf{u}(t) \in \Re^m$

Parameters: $\mathbf{p}(t) \in \Re^\ell$

Output vector: $\mathbf{y}(t) \in \Re^r$



Bolza Type Cost Function Minimization:

$$J = \phi\left[\mathbf{x}(t_f), t_f\right] + \int_{t_0}^{t_f} \mathbf{L}\left[\mathbf{x}(\tau), \mathbf{u}(\tau), \tau\right] d\tau$$

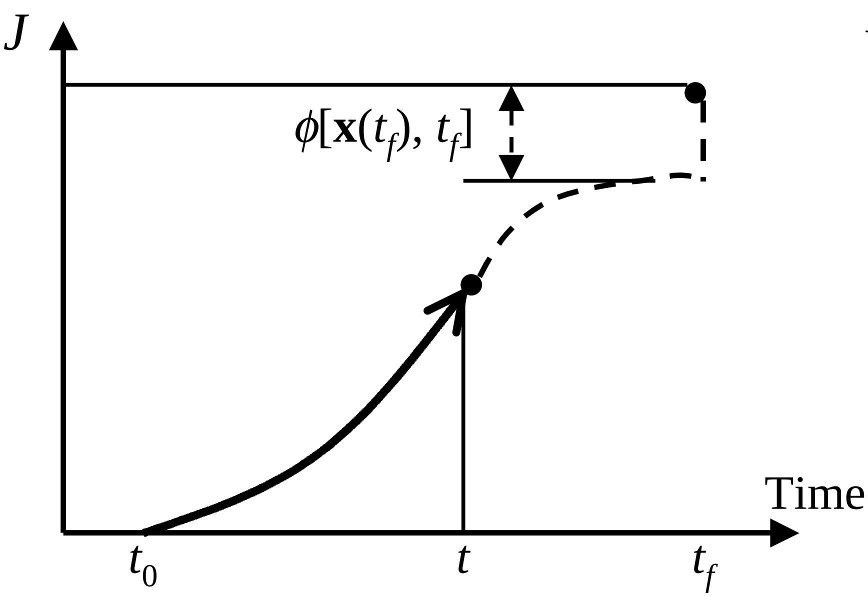
Value Function Minimization

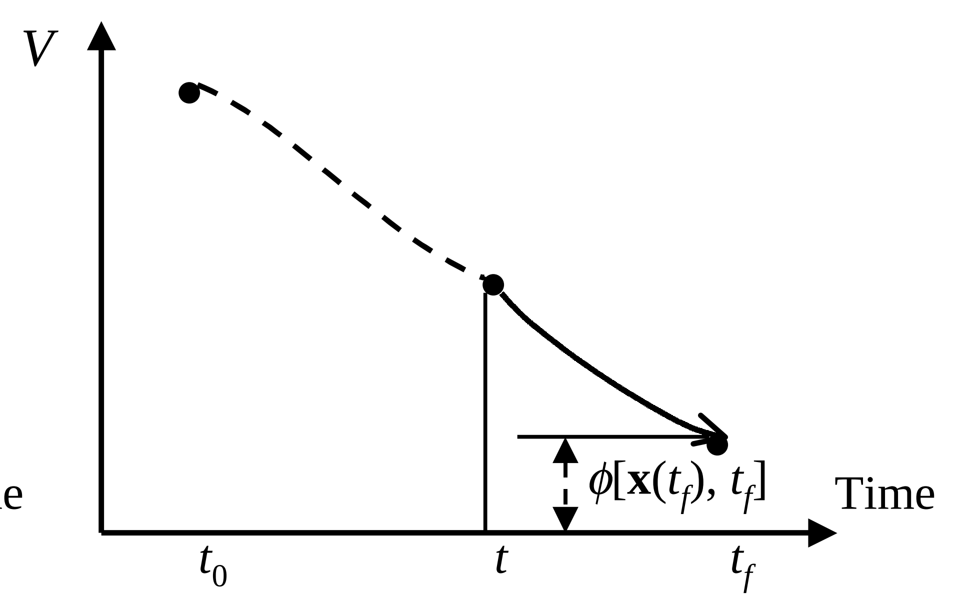
Value Function for [t, t_f]:

$$V\left[\underset{t \le \tau \le t_f}{\mathbf{x}(t), \mathbf{u}(\tau)}, t\right] = \phi\left[\mathbf{x}(t_f), t_f\right] + \int_t^{t_f} \mathbf{L}\left[\mathbf{x}(\tau), \mathbf{u}(\tau), \tau\right] d\tau$$

The minimization of J can be ***imbedded*** in the following problem:

$$V^*\left[\mathbf{x}(t), t\right] = \min_{\substack{\mathbf{u}(\tau) \\ t \le \tau \le t_f}} \left\{ \phi\left[\mathbf{x}(t_f), t_f\right] + \int_t^{t_f} \mathbf{L}\left[\mathbf{x}(\tau), \mathbf{u}(\tau), \tau\right] d\tau \right\}$$





Discretized Optimization Problem

Approximate the equations of motion by a difference equation:

$$\mathbf{x}(k+1) = \mathbf{f}_D[\mathbf{x}(k), \mathbf{p}(k), \mathbf{u}(k)],$$

where: $\mathbf{x}(k) \equiv \mathbf{x}(k\Delta t), \quad \Delta t = t_f / k_f, \quad k \rightarrow t_k = 0, \Delta t, \ldots, (k_f - 1)\Delta t, k_f \Delta t$

Similarly, the cost function:

$$J = \phi[\mathbf{x}(k_f), k_f] + \sum_{k=0}^{k_f-1} \mathbf{L}_D[\mathbf{x}(k), \mathbf{u}(k), k]$$

The cost of operation during the last stage is:

$$V_{k_f-1,k_f}[\mathbf{x}(k_f-1), \mathbf{u}(k_f-1)] = \phi[\mathbf{x}(k_f), k_f] + \mathbf{L}_D[\mathbf{x}(k_f-1), \mathbf{u}(k_f-1)]$$

The Principle of Optimality

The optimal cost, from t_{k_f-1} to t_{k_f}, is then:

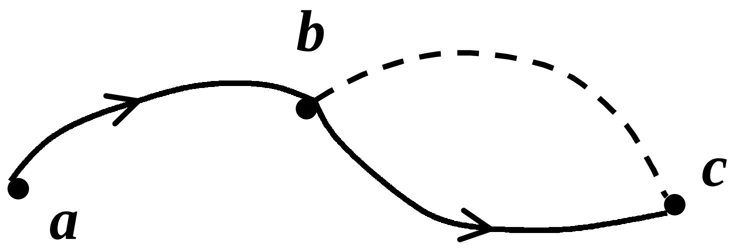
$$V^*_{k_f-1,k_f}\left[\mathbf{x}(k_f-1)\right] \equiv \min_{\mathbf{u}(k_f-1)}\left\{\phi\left[\mathbf{x}(k_f),k_f\right]+\mathbf{L}_D\left[\mathbf{x}(k_f-1),\mathbf{u}(k_f-1)\right]\right\}$$

Similarly, the optimal cost over the last two intervals is given by:

$$V^*_{k_f-2,k_f}\left[\mathbf{x}(k_f-2)\right] \equiv$$

$$\min_{\mathbf{u}(k_f-2),\mathbf{u}(k_f-1)}\left\{\mathbf{L}_D\left[\mathbf{x}(k_f-2),\mathbf{u}(k_f-2)\right]+V_{k_f-1,k_f}\left[\mathbf{x}(k_f-1),\mathbf{u}(k_f-1)\right]\right\}$$

By the principle of optimality,



Given V_{ab},

if V_{abc} is optimal from ***a*** to ***c***,

then V_{bc} is optimal from ***b*** to ***c***:

$$V^*_{abc} = V_{ab} + V^*_{bc}$$

A Recurrence Relationship of Dynamic Programming

So, the optimal cost for a 2-stage process can be re-written as:

$$V^*_{k_f-2,k_f}\left[\mathbf{x}\left(k_f-2\right)\right]=\min_{\mathbf{u}\left(k_f-2\right)}\left\{\mathbf{L}_D\left[\mathbf{x}\left(k_f-2\right),\mathbf{u}\left(k_f-2\right)\right]+V^*_{k_f-1,k_f}\right\}$$

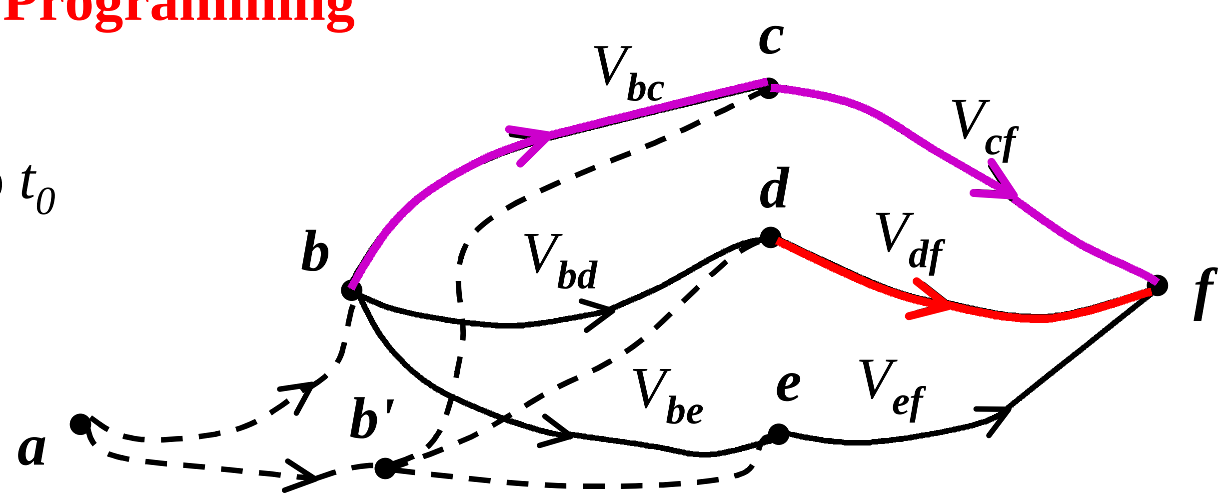
Then, for a k-stage process:

$$V^*_{k_f-n,k_f}\left[\mathbf{x}\left(k_f-n\right)\right]\equiv\min_{\mathbf{u}\left(k_f-n\right),\mathbf{u}\left(k_f-n+1\right),\ldots,\mathbf{u}\left(k_f-1\right)}\left\{\phi\left[\mathbf{x}\left(k_f\right),k_f\right]+\sum_{k=k_f-n}^{k_f-1}\mathbf{L}_D\left[\mathbf{x}(k),\mathbf{u}(k)\right]\right\}$$

$$=\min_{\mathbf{u}\left(k_f-n\right)}\left\{\mathbf{L}_D\left[\mathbf{x}\left(k_f-n\right),\mathbf{u}\left(k_f-n\right)\right]+V^*_{k_f-n+1,k_f}\right\}$$

Backward Dynamic Programming

- Begin at t_f
- Move backward to t_0
- Store all $\mathbf{u}$ and V
- **Off-line process**



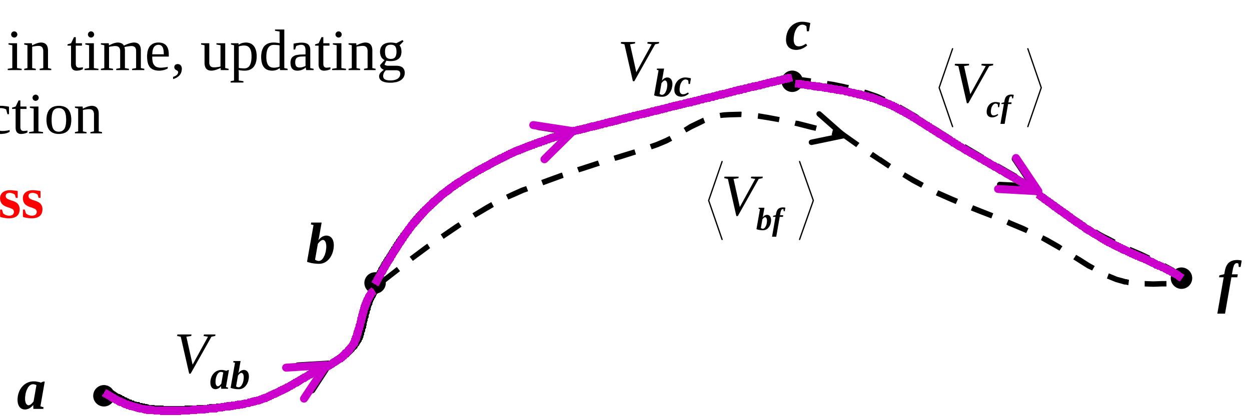
Forward Dynamic Programming

Cost associated with going from t_k to t_f:

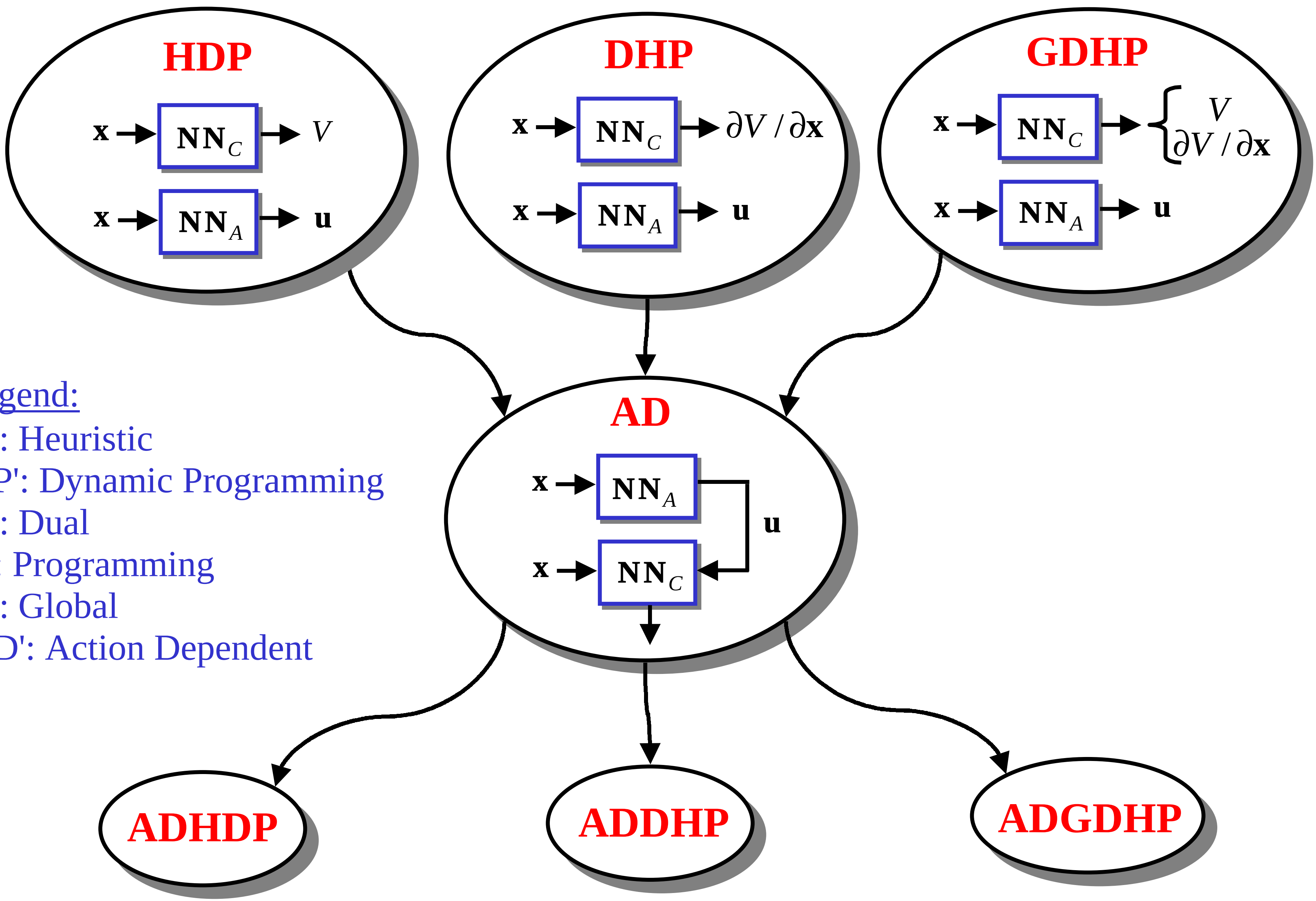
$$V[\mathbf{x}(t_k)] \equiv V_{k,k_f} = \underbrace{U\{\mathbf{x}(t_k), \mathbf{u}[\mathbf{x}(t_k)]\}}_{\text{Utility}} + \underbrace{\langle V[\mathbf{x}(t_{k+1})]\rangle}_{\text{Estimated cost for } t_{k+1} \le t \le t_f}$$

Hereon, let $t = t_k$, $t + 1 = t_{k+1}$, etc...

- Estimate cost-to-go function, $\langle \bullet \rangle$
- Determine immediate control action
- Move forward in time, updating cost-to-go function
- **On-line process**



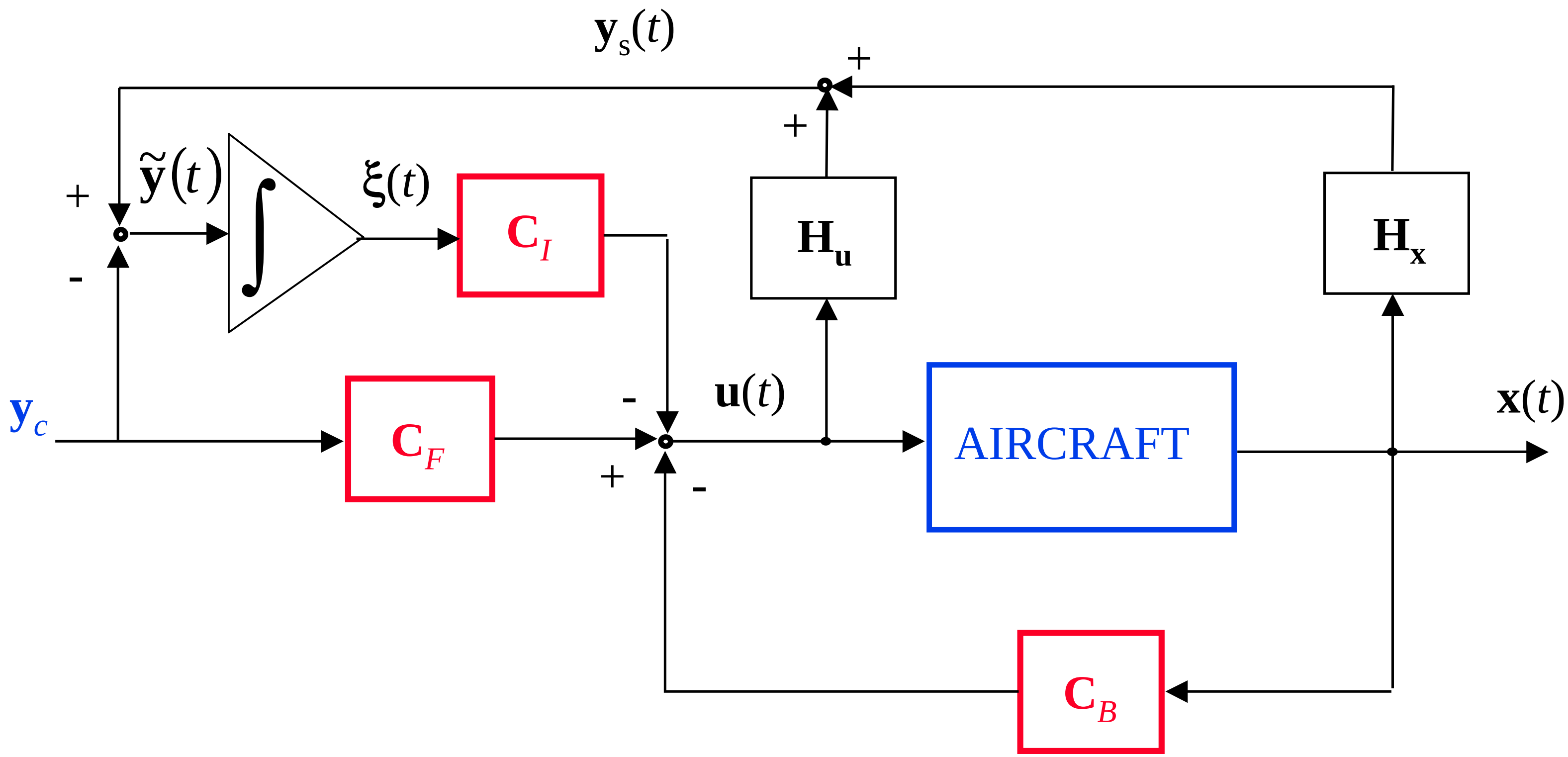
Adaptive Critic Designs .. at a Glance!



Legend:
'H': Heuristic
'DP': Dynamic Programming
'D': Dual
'P': Programming
'G': Global
'AD': Action Dependent

Proportional-Integral Controller

Closed-loop stability: $\mathbf{x}(t) \to \mathbf{x}_c, \quad \mathbf{u}(t) \to \mathbf{u}_c, \quad \tilde{\mathbf{y}}(t) \to 0$



Omitting Δ's, for simplicity:

$\tilde{\mathbf{y}}(t) = \mathbf{y}_S(t) - \mathbf{y}_C, \quad \tilde{\mathbf{u}}(t) = \mathbf{u}(t) - \mathbf{u}_C, \ldots, \qquad \mathbf{y}_c$ = desired output, $(\mathbf{x}_c, \mathbf{u}_c)$ = set point.

Linearized Business Jet Aircraft Model

Assuming small perturbations, expand about nominal solution:

$$\Delta\dot{\mathbf{x}}(t) = \mathbf{F}\Delta\mathbf{x}(t) + \mathbf{G}\Delta\mathbf{u}(t)$$
$$\Delta\mathbf{y}(t) = \mathbf{H_x}\Delta\mathbf{x}(t) + \mathbf{H_u}\Delta\mathbf{u}(t)$$

where: $\mathbf{F} \equiv \dfrac{\partial\mathbf{f}(\mathbf{x},\mathbf{u})}{\partial\mathbf{x}}, \quad \mathbf{G} \equiv \dfrac{\partial\mathbf{f}(\mathbf{x},\mathbf{u})}{\partial\mathbf{u}}, \quad \mathbf{H_x} \equiv \dfrac{\partial\mathbf{h}(\mathbf{x},\mathbf{u})}{\partial\mathbf{x}}, \quad \mathbf{H_u} \equiv \dfrac{\partial\mathbf{h}(\mathbf{x},\mathbf{u})}{\partial\mathbf{u}}$

The state variable is augmented to include the output error integral, $\boldsymbol{\xi}(t)$,

$$\Delta\dot{\mathbf{x}}_a(t) \equiv \left[\Delta\dot{\tilde{\mathbf{x}}}^T(t) \quad \Delta\dot{\boldsymbol{\xi}}^T(t)\right]^T = \begin{bmatrix} \mathbf{F} & \mathbf{0} \\ \mathbf{H_x} & \mathbf{0} \end{bmatrix} \Delta\mathbf{x}_a(t) + \begin{bmatrix} \mathbf{G} \\ \mathbf{H_u} \end{bmatrix} \Delta\tilde{\mathbf{u}}(t)$$

and the **Proportional-Integral cost function** is defined as:

$$J = \frac{1}{2}\int_0^\infty \left[\Delta\mathbf{x}_a^T(\tau)\mathbf{Z}\Delta\mathbf{x}_a(\tau) + 2\Delta\mathbf{x}_a^T(\tau)\mathbf{S}\Delta\tilde{\mathbf{u}}(\tau) + \Delta\tilde{\mathbf{u}}^T(\tau)\mathbf{R}\Delta\tilde{\mathbf{u}}(\tau)\right]d\tau$$

Linear Proportional-Integral Control Law Formulation

From the Euler-Lagrange equations, the **optimal*** **control law** is:

$$\Delta\tilde{\mathbf{u}}^*(t) = -\mathbf{R}^{-1}\left[\mathbf{G}^T\mathbf{P} + \mathbf{M}^T\right]\Delta\mathbf{x}_a(t)$$
$$= -\mathbf{C}\Delta\mathbf{x}_a(t) = -\left[\mathbf{C}_B \quad \mathbf{C}_I\right]\Delta\mathbf{x}_a(t)$$

where **P** is a *Riccati matrix*.

Furthermore,

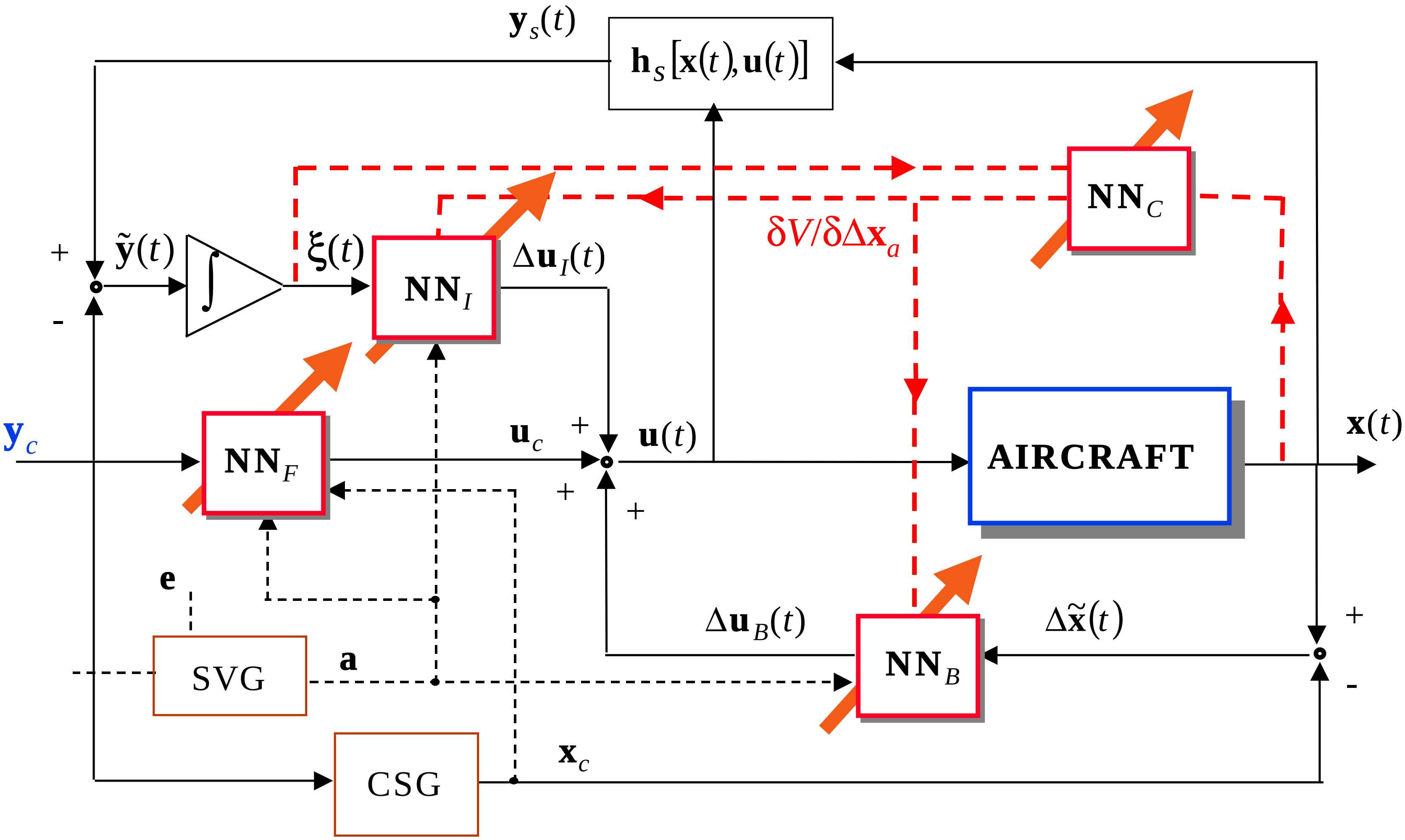
it can be shown that the **optimal value function** is

$$V^*\left[\Delta\mathbf{x}_a(t)\right] = \frac{1}{2}\Delta\mathbf{x}_a^T(t)\mathbf{P}\Delta\mathbf{x}_a(t)$$

and its partial derivative with respect to $\Delta\mathbf{x}_a$ is:

$$\frac{\partial V^*}{\partial \Delta\mathbf{x}_a}\left[\Delta\mathbf{x}_a(t)\right] = \Delta\mathbf{x}_a^T(t)\mathbf{P}$$

Proportional-Integral Neural Network Controller



Where: $\mathbf{x}(t)\rightarrow\mathbf{x}_c,\quad \mathbf{u}(t)\rightarrow\mathbf{u}_c,\quad \tilde{\mathbf{y}}(t)\rightarrow 0,\quad \mathbf{y}_s(t)\rightarrow\mathbf{y}_c$

Feedback and Command-Integral (Action) Neural Network Pre-Training

Feedback Requirements (pre-training phase):

$\mathbf{NN}_B$ accounts for regulation, $\Delta\mathbf{u}_B = \mathbf{NN}_B(\Delta\tilde{\mathbf{x}}, \mathbf{a})$.

For each operating point, k,

a $\mathbf{z}$ output and inputs:

$$\mathbf{p} \equiv \begin{bmatrix} \mathbf{p}_{\Delta\tilde{\mathbf{x}}}^T & \mathbf{p}_{\mathbf{a}}^T \end{bmatrix}^T \longrightarrow$$

(R1) $\mathbf{z}(\mathbf{0}, \mathbf{a}^k) = \mathbf{0}$

(R2) $\left.\dfrac{\partial \mathbf{z}}{\partial \mathbf{p}_{\Delta\mathbf{x}}}\right|_k = \left.\dfrac{\partial(\Delta\mathbf{u}_B)}{\partial(\Delta\mathbf{x})}\right|_k = -\mathbf{C}_B(\mathbf{a}^k)$

Command-Integral Requirements (pre-training phase):

$\mathbf{NN}_I$ provides dynamic compensation, $\Delta\mathbf{u}_I = \mathbf{NN}_I(\Delta\xi, \mathbf{a})$.

For each operating point, k,

a $\mathbf{z}$ output and inputs:

$$\mathbf{p} \equiv \begin{bmatrix} \mathbf{p}_{\Delta\xi}^T & \mathbf{p}_{\mathbf{a}}^T \end{bmatrix}^T \longrightarrow$$

(R1) $\mathbf{z}(\mathbf{0}, \mathbf{a}^k) = \mathbf{0}$

(R2) $\left.\dfrac{\partial \mathbf{z}}{\partial \mathbf{p}_{\Delta\xi}}\right|_k = \left.\dfrac{\partial(\Delta\mathbf{u}_I)}{\partial(\Delta\xi)}\right|_k = -\mathbf{C}_I(\mathbf{a}^k)$

Critic Neural Network Pre-Training

From the Proportional-Integral optimal value function derivatives:

- $\boldsymbol{\lambda}[\Delta\mathbf{x}_a(t)] \equiv \dfrac{\partial V^*}{\partial \Delta\mathbf{x}_a}[\Delta\mathbf{x}_a(t)] = \Delta\mathbf{x}_a^{\ T}(t)\mathbf{P} \;\rightarrow\; \boldsymbol{\lambda}[\mathbf{0}] = \mathbf{0}$
- $\dfrac{\partial^2 V^*}{\partial(\Delta\mathbf{x}_a)^2}(t) = \mathbf{P}$

Critic Requirements (pre-training phase):

$\mathbf{NN}_C$ estimates the value function derivatives, $\boldsymbol{\lambda} = \mathbf{NN}_C(\Delta\mathbf{x}_a, \mathbf{a})$.

For each operating point, k,
a $\mathbf{z}$ output and training inputs:

$$\mathbf{p} \equiv \left[\mathbf{p}_{\Delta\mathbf{x}_a}^{\ T} \quad \mathbf{p}_{\mathbf{a}}^{\ T}\right]^T \longrightarrow$$

(R1) $\mathbf{z}(\mathbf{0}, \mathbf{a}^k) = 0$

(R2) $\left.\dfrac{\partial \mathbf{z}}{\partial \mathbf{p}_{\Delta\mathbf{x}_a}}\right|_k = \left.\dfrac{\partial^2 V^*}{\partial(\Delta\mathbf{x}_a)^2}\right|_k = \mathbf{P}(\mathbf{a}^k)$

Action and Critic Neural Network On-line Training by Dual Heuristic Programming

The same cost function is optimized in pre-training and in on-line learning

$$J = \frac{1}{2}\sum_{t=0}^{\infty}\left[\Delta\mathbf{x}_a^T(t)\mathbf{Z}\Delta\mathbf{x}_a(t) + 2\Delta\mathbf{x}_a^T(t)\mathbf{S}\Delta\tilde{\mathbf{u}}(t) + \Delta\tilde{\mathbf{u}}^T(t)\mathbf{R}\Delta\tilde{\mathbf{u}}(t)\right]$$

The cost-to-go at time *t*,

$$V[\Delta\mathbf{x}_a(t)] = U\{\Delta\mathbf{x}_a(t), \Delta\tilde{\mathbf{u}}[\Delta\mathbf{x}_a(t)]\} + \langle V[\Delta\mathbf{x}_a(t+1)]\rangle,$$

must be minimized w.r.t. $\Delta\tilde{\mathbf{u}}(t)$, for an estimated cost-to-go at (t + 1).

The utility function is defined as:

$$U[\Delta\mathbf{x}_a(t), \Delta\tilde{\mathbf{u}}(t)] = \frac{1}{2}\left[\Delta\mathbf{x}_a^T(t)\mathbf{Z}\Delta\mathbf{x}_a(t) + 2\Delta\mathbf{x}_a^T(t)\mathbf{S}\Delta\tilde{\mathbf{u}}(t) + \Delta\tilde{\mathbf{u}}^T(t)\mathbf{R}\Delta\tilde{\mathbf{u}}(t)\right]$$

Optimality Conditions for Dual Heuristic Programming

Differentiating both sides of the value function, $V[\Delta\mathbf{x}_a(t)]$, w.r.t. $\Delta\mathbf{x}_a(t)$:

$$\boldsymbol{\lambda}[\Delta\mathbf{x}_a(t)] \equiv \frac{\partial V[\Delta\mathbf{x}_a(t)]}{\partial \Delta\mathbf{x}_a(t)} = \frac{\partial U[\bullet]}{\partial \Delta\mathbf{x}_a(t)} + \frac{\partial U[\bullet]}{\partial \Delta\tilde{\mathbf{u}}(t)} \frac{\partial \Delta\tilde{\mathbf{u}}[\Delta\mathbf{x}_a(t)]}{\partial \Delta\mathbf{x}_a(t)} + \left\langle \boldsymbol{\lambda}[\Delta\mathbf{x}_a(t+1)] \frac{\partial \Delta\mathbf{x}_a(t+1)}{\partial \Delta\mathbf{x}_a(t)} \right\rangle +$$

$$\left\langle \boldsymbol{\lambda}[\Delta\mathbf{x}_a(t+1)] \frac{\partial \Delta\mathbf{x}_a(t+1)}{\partial \Delta\tilde{\mathbf{u}}(t)} \frac{\partial \Delta\tilde{\mathbf{u}}[\Delta\mathbf{x}_a(t)]}{\partial \Delta\mathbf{x}_a(t)} \right\rangle \equiv \mathbf{F}_c(\bullet),$$

Optimality equation:

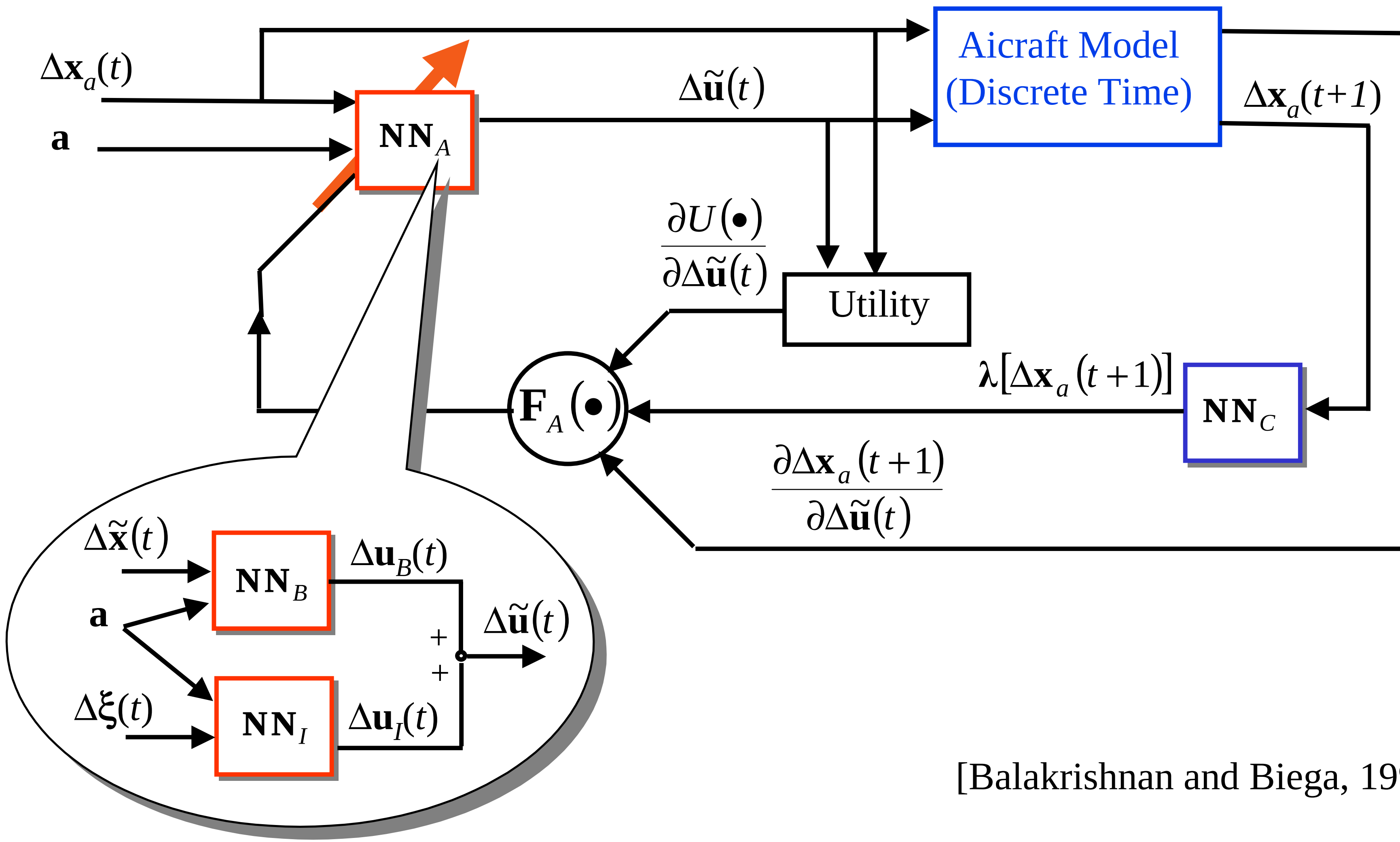
$$\boxed{\frac{\partial V[\Delta\mathbf{x}_a(t)]}{\partial \Delta\tilde{\mathbf{u}}(t)} = \mathbf{0}}$$

Differentiating w.r.t. the control, $\Delta\tilde{\mathbf{u}}(t)$:

$$\frac{\partial U[\bullet]}{\partial \Delta\tilde{\mathbf{u}}(t)} + \left\langle \boldsymbol{\lambda}[\Delta\mathbf{x}_a(t+1)] \frac{\partial \Delta\mathbf{x}_a(t+1)}{\partial \Delta\tilde{\mathbf{u}}(t)} \right\rangle \equiv \mathbf{F}_A(\bullet) = \mathbf{0}$$

Action Network On-line Training

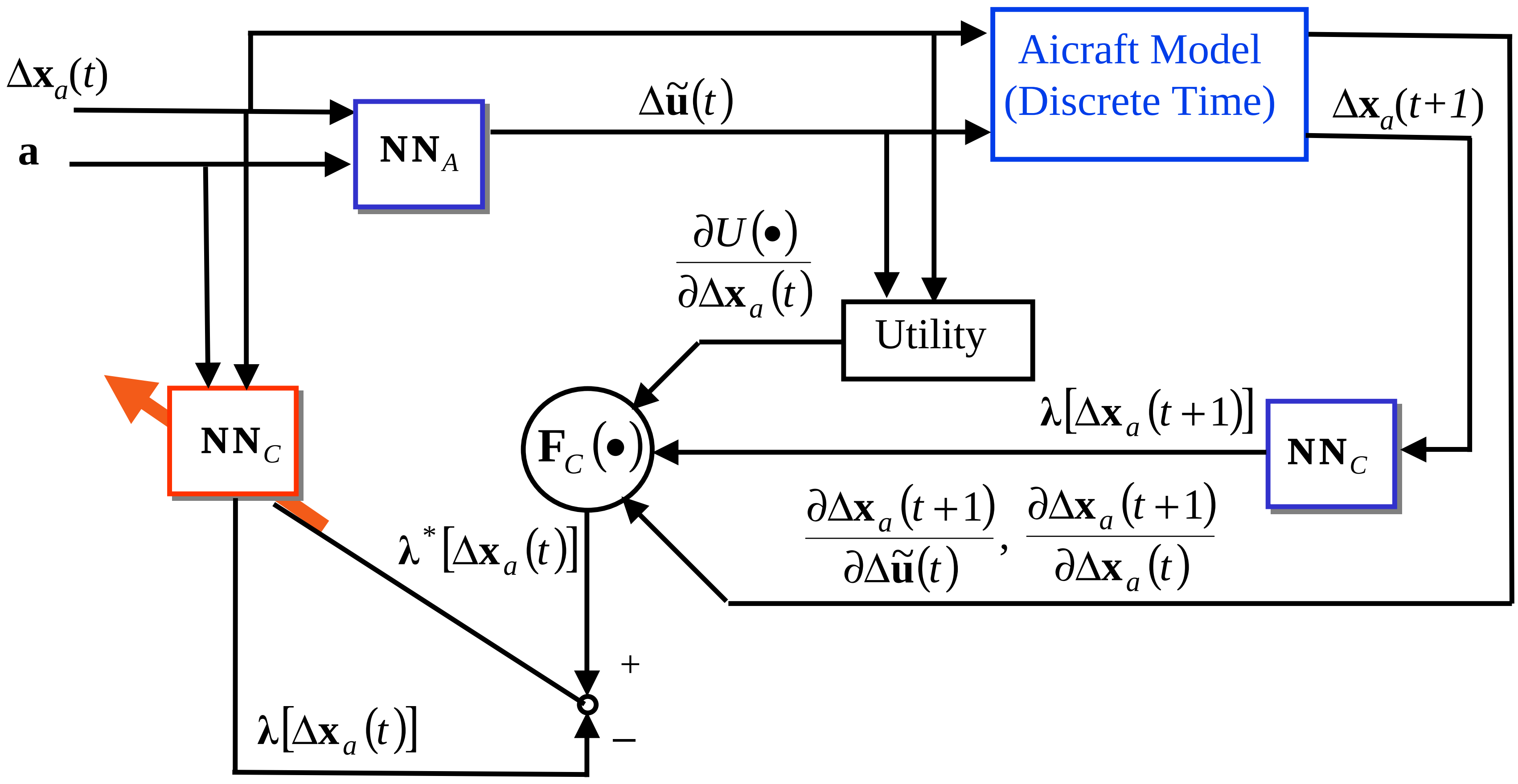
Train action network, at time *t*, holding the critic parameters fixed



[Balakrishnan and Biega, 1996]

Critic Network On-line Training

Train critic network, at time t, holding the action parameters fixed



[Balakrishnan and Biega, 1996]

Summary

- Aircraft optimal control problem
- Linear multivariable control structure
- Corresponding nonlinear controller
- Adaptive critic architecture:

  Action and critic networks

  - Algebraic pre-training based on a-priori knowledge
  - On-line training during simulations (severe conditions)

Conclusions and Future Work

- Nonlinear, adaptive, real-time controller:
  Maintain stability and robustness throughout flight envelope
  Improve aircraft control performance under extreme conditions

- Systematic approach for designing nonlinear control systems motivated by well-known linear control structures

- Innovative neural network training techniques

Future Work:

- Adaptive critic architecture implementation

- Testing: acrobatic maneuvers, severe operating conditions,
  coupling and nonlinear effects